Marked-up Specifcation

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
APPLICATION FOR LETTERS PATENT

Title : INFORMATION PROCESSING UNIT WITH PARALLEL PROCESSOR ON A CHIP CAPABLE OF INDEPENDENT OPERATION

Inventor(s) : Makoto OGAWA
Tadashi SHIBATA

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is based upon and claims the benefit of priority from the prior Japanese Patent Application No. 2003-039741, filed on February 18, 2003, the entire contents of which are incorporated herein by reference.

BACKGROUND OF THE INVENTION

~~[~~Field of the Invention~~]~~

[0002] The present invention relates to an information processing unit, particularly, the present invention is suitable for ~~applying~~application to an information processing unit having a plurality of processors on one chip, which are able to operate in parallel ~~in one chip~~.

~~[~~Description of the Related Art~~]~~

[0003] ~~There exists a~~An SIMD-(Single Instruction Multiple Data)-parallel processor ~~as~~is one type of parallel ~~computers~~computer. The SIMD parallel processor has a plurality of processors ~~in~~on one chip, executing ~~one processing~~a process corresponding to ~~a supplied~~an instruction supplied thereto in parallel simultaneously ~~by~~with the plural processors.

[0004] Each processor in the SIMD parallel process consists of a processor element and a memory. In the SIMD processor, ~~as one processing~~since a process corresponding to the supplied instruction is to be ~~operated~~carried out in parallel by all processors, the memory of each processor holds only data.

[0005] Therefore, though the SIMD processor has ~~the~~ plural processors, it is not able to execute different kinds of processing ~~by~~with each processor in parallel. For example, while executing a retrieval ~~processing by~~process with some processors, the SIMD processor can not execute a detailed comparing ~~processing by~~process with other processors in parallel, in a processing of matching data and so on.

[0006] In order to execute different kinds of processing in parallel ~~by~~with the plural processors, it is necessary that the program (instruction) should be ~~held~~stored in the memory of each processor. However, in ~~an~~the instruction set used for ~~the~~a general-purpose processor in recent years, a single instruction is organized ~~by~~into 32-~~bit~~bits or 64-~~bit~~bits. If the memory of each processor holds the program by which the same processing as ~~the~~done in a general-purpose processor ~~execute~~is executed, high memory capacity is required. However, in order to provide many processors in one chip, the memory capacity of each processor is ~~preferable~~preferred to be small.

SUMMARY OF THE INVENTION

[0007] In view of the above, it is an object of the present invention to enable a plurality of processors provided ~~in~~on one chip to execute various and different kinds of processing in parallel, while ~~lowering~~minimizing the increase of the memory capacity required to ~~hold~~store a program in each processor.

[0008] An information processing unit according to the present invention is an information processing unit having a plurality of processors ~~in~~on one chip, which is characterized by ~~that the~~ plural processors that can execute ~~the~~ instructions independently. Each processor includes a decoder circuit for determining the instruction to be ~~executed~~ uniquely executed, based on an input history of instruction codes, by use of an instruction code ~~inputted~~input to the processor from a plurality of instructions which are assigned to the instruction code.

[0009] According to the present invention, ~~as~~since one instruction code corresponds to ~~the~~a plurality of instructions, the instruction code length can be shorter than the case ~~that~~where different instruction codes are made to correspond to the plurality of instructions, so that different kinds of processing can be performed by the same instruction code.

BRIEF DESCRIPTION OF THE DRAWINGS

[00010] Fig.1 is a block diagram showing a configuration of a parallel processor to which an information processing unit ~~applied~~ according to an embodiment of the present invention is applied;

[00011] Fig.2 is a view explaining the concept of the processing operation in each processor according to the present embodiment;

[00012] Fig.3 is a block diagram showing a configuration of an instruction decoding portion;

[00013] Fig.4 is a view showing an example of a look up table for setting ~~the~~a rule for changing a group code; and

[00014] Fig.5 is a view showing an example of a group change determination circuit.

DETAILED DESCRIPTION OF THE ~~PREFERRED EMBODIMENTS~~INVENTION

[00015] An embodiment of the present invention will be described hereinafter in reference to the drawings.

[00016] Fig.1 is a block diagram showing a configuration of a parallel processor 10 to which the information processing unit of ~~the~~an embodiment according to the present invention applies.

[00017] The parallel processor 10 is composed of plural processors 11 which are connected to each other ~~in~~and contained on one chip. Each processor 11 has a memory 12, an instruction decoding portion 13 and a plurality of processor elements (PE) 14, each of which is able to execute a certain instruction independently. The memory 12 includes a program memory (instruction memory area) for storing a program (instruction) and a data memory (data memory area) for storing data.

[00018] The instruction decoding portion 13 decodes the instruction, reading out of the program memory portion of the memory 12. Also the instruction decoding portion 13 controls an internal ~~resister~~register described later and the processor elements 14 based on the result of decoding. ~~Detailed~~A detailed description of the instruction decoding portion 13 will be given later. The processor elements 14 are ~~the~~ conventionally known SIMD processor elements, executing operations according to control signals supplied from the instruction decoding portion 13.

[00019] Though the processors 11 each have 8-processor elements 14 as ~~one~~in the example shown in Fig.1, each processor 11 is able to have optional numbers of processor elements 14, for example, a single processor element will be acceptable.

[00020] Fig.2 is a view explaining the concept of the processing operation in each processor 11. In the following description, one instruction code is organized ~~with~~to have 8-~~bit~~bits, and a group code described later is organized ~~with~~to have 3-~~bit~~bits.

[00021] In Fig.2, 21 is an instruction queue, being composed of plural instruction memories so as to hold plural instruction codes. Each instruction memory has an 8-bit memory area.

[00022] The instruction decoding portion 13 includes a group register 22 and an instruction decoder circuit 23. The group register 22 is a register for storing a group code, having 3-bit memory area. The instruction decoder circuit 23 decodes the instruction code, ~~referring~~with reference to the group code to execute a control operation according to the decoded result.

[00023] The elements 24-$_i$ ~~(~~, where "i" is a subscript, i = 1 ~ n (n is an

optional natural number)~~]~~, are processor elements equivalent to the processor elements 14 shown in Fig.1.

[00024] The group code designates one of the groups into which the input instructions concerning the processing executable in the processor 11 are sorted based on a predetermined rule. The group codes ~~are~~have different ~~value~~values from each other in every group.

[00025] The input instructions are sorted into groups, such as the instruction ~~of~~for the four basic operations ~~like~~of addition, subtraction, multiplication, and division ~~including the~~. They also include a shift operation or ~~the~~a bit operation, and ~~the~~an instruction ~~of the~~for a WTA operation used for matching data and the like, ~~the~~a burst operation instruction, ~~the~~a register operation instruction, ~~the~~a memory control instruction, ~~the~~a flow control instruction, ~~the~~a scalar instruction and so on. The instruction codes of respective input instructions are different from each other in one single group, ~~and~~but the same instruction code can exist in another group. Namely, plural instructions from different groups can be expressed by one instruction code.

[00026] ~~Besides~~ In addition, there ~~exists a~~exist global ~~instruction~~instructions which does not belong to any group (which ~~is~~are independent of the groups). The global instructions are, for example, a group change instruction for changing the group code, a group temporary change instruction for changing the group code of immediately following instruction, an alias execution instruction for executing an optional instruction previously assigned and the like.

[00027] The operation will be described as follows.

[00028] The instruction decoder circuit 23 read an instruction code ICD~~ 1 ~~, one byte (8-bits) at a time out of the instruction memory of the instruction queue 21. The instruction decoder circuit 23 adds a group code GR supplied from the group register 22 to the read-out instruction code ICD and decodes them. Namely, the instruction decoder circuit 23 decodes an 11-bit code (hereinafter referred to as ~~'~~the "internal instruction code~~'~~") consisting of the 3-bit group code GR and the 8-bit instruction code.

[00029] When the instruction is a standard instruction, for example, ~~the~~an instruction for one of the four basic operations and the like as ~~one~~a result of decoding, the instruction decoder circuit 23 supplies a control signal INS corresponding to the instruction to each of the processor elements 24-$_{i}$. When the instruction is to change the group code, such as the group (temporary) change instruction as ~~another~~a result of decoding, the

instruction decoder circuit 23 writes the thus-changed group code GRS designated by this instruction into the group register 22.

[00030] Fig.3 is a block diagram showing an example of a configuration ~~example~~ of the instruction decoding portion 13.

[00031] In Fig.3, numeral 31 ~~is~~designates an instruction memory for holding the instruction code~~, 32 is~~; processor elements 32 are used for executing various kinds of processing according to ~~the~~ control by the instruction decoding portion 13.

[00032] The instruction decoding portion 13 includes an instruction bit linkage part 33, an alias instruction decoder 34, an alias instruction register $35_{\_j}$ ~~(~~, where "j" is a subscript and an optional natural number~~)~~, a first selector 36, a group change instruction decoder 37, a standard instruction decoder 38, an alias change instruction decoder 39, a plural-cycle instruction state register 40, a group register 41, a second selector 42, and a group temporary memory register 43.

[00033] The instruction bit linkage part 33 reads the instruction code out of the instruction memory 31 ~~1~~ one byte at a time. This instruction bit linkage part 33 is ~~defined~~configured, based on the instruction code which ~~is inputted~~has been input in the past, so as to output ~~the~~an internal instruction code, ~~with~~by linking (adding) of the group code held in the group temporary memory register 43 to the instruction code read out of the instruction memory 31.

[00034] The alias instruction decoder 34 determines whether the internal instruction code supplied from the instruction bit linkage part 33 is an alias execution instruction which replaces the internal instruction code with the code of the alias instruction, or not. Depending on the determination, when the internal instruction code is ~~the~~an alias execution instruction, the alias instruction decoder 34 reads the alias instruction (the internal instruction to be replaced) out of the alias instruction register 35_{-j} corresponding to the group code of the internal instruction code so as to output the instruction.

[00035] The alias instruction registers 35_{-j} are set in every group, in which the internal instruction codes of the instruction of other groups assigned as the alias instruction are stored respectively. As for the alias instruction registers 35_{-j}, they are not ~~only~~ limited to only those in the embodiment shown in Fig.3, but also to another configuration that has a common alias instruction ~~in~~for all groups, or a still another configuration that has plural alias instructions in each group.

[00036] The first selector 36 selectively outputs the internal instruction code supplied from the instruction bit linkage part 33, or the internal instruction code supplied from the

alias instruction decoder 34.

[00037] The group change instruction decoder 37 changes one group code into another new group code, when the internal instruction code supplied from the first selector 36 is ~~the~~a group change instruction or ~~the~~a group temporary change instruction. In addition to the group (temporary) change instruction, the group change instruction decoder 37 can change the group code into a new group code when the internal instruction code supplied from the first selector 36 satisfies the rule for changing the group code which has been previously defined. The rule for changing the group code can be defined~~,~~ with, for example, a look up table (LUT) 44, which relates the internal instruction code ~~with~~to the group code to be changed~~, being~~. The look up table is provided in the group change instruction decoder 37.

[00038] The group change instruction decoder 37 writes a new group code value into the group ~~resister~~register 41 in the case of changing the group code based on the group change instruction or the rule for changing the group code. Hereby, the value held in the group register 41 is replaced with the new group code value. Further, the value held in the group register 41 is supplied to the group temporary memory register 43 through the second selector 42 to be held therein.

[00039] When only the group code of an immediately following instruction is to be changed based on the group temporary change instruction, the group change instruction decoder 37 supplies the subject group code value to the group temporary memory register 43 through the second selector 42. Hereby, the value held in the group temporary memory register 43 is replaced with a new group code value. In this case, the group change instruction decoder 37 does not write the group code into the group register 41.

[00040] The standard instruction decoder 38 decodes the internal instruction code and outputs ~~the~~a control signal to the processor elements 32 in the same way as ~~the well-known~~in a conventional processor, when the internal instruction code supplied from the first selector is a standard instruction (such as the four basic ~~operation~~operations).

[00041] The alias change instruction decoder 39 rewrites the value of the alias instruction register 35_{-j}, when the internal instruction code supplied from the first selector 36 is ~~the~~an alias change instruction code. Specifically, the alias change instruction decoder 39 rewrites the value of the alias instruction register 35_{-j} corresponding to the group code ~~whereof~~, the change of which is indicated by the internal instruction code, into the internal instruction code (the group code as well as the instruction code) of the instruction code to be replaced according to the internal instruction.

[00042] The plural-cycle instruction state register 40 controls, when the alias instruction or the like replaced by the alias execution instruction is ~~the~~an instruction which occupies more than a 2-byte length, the first selector 36 to replace only the first ~~1-~~byte of the instruction with the alias instruction and ignore the alias instruction over ~~2-~~the second byte.

[00043] The plural-cycle instruction state register 40 therefore controls the first selector 36 so as to only select ~~1-~~the first byte of the internal instruction code from the alias instruction decoder 34 and subsequently select the internal instruction code from the instruction bit linkage part 33 over ~~2-byte~~the two bytes. That is why the first ~~1-~~byte of the instruction indicates the instruction itself and the instruction over ~~2-~~two byte may be data or the like, when the alias instruction or the like has more than ~~2-~~a two byte length.

[00044] Incidentally, the group change instruction decoder 37, the standard instruction decoder 38, the alias change instruction decoder 39 and the plural-cycle instruction state register 40 communicate with each other ~~for~~during operation.

[00045] The second selector 42 selectively supplies the group code value held in the group register 41 or the group code value supplied from the group change instruction decoder 37 to the group temporary memory register ~~43 selectively,~~43, according to the control of the group change instruction decoder 37. Specifically, the second selector 42 selects and outputs the value from the group change instruction decoder 37 only in the case ~~that~~where the group code is changed according to the group temporary change instruction. Otherwise, the second selector 42 will select and output the value from the group register 41.

[00046] Though the instruction decoding portion 13 ~~does~~is not ~~include~~shown with a flow control circuit and the like such as a program counter, a register, a conditional branch, those can be provided therein.

[00047] Fig.4 is an explanatory view showing an example of the LUT 44, which sets the rule for changing the group code.

[00048] As shown in Fig.4, the rule for changing the group code, as one group, is to be defined with an instruction mask IM_k ~~(~~, where "k" is a subscript and optional natural number~~ and, the followings are the same condition)~~. The following two entries, an instruction code IC_k and a changed group code GC_k have the same subscript. The instruction mask IM_k and the instruction code IC_k are defined by 11-bit equivalent to the group code (3-bit) and the instruction code (8-bit), and the group code GC_k is defined by the group code only (3-bit).

[00049] The instruction mask IM_k sets a bit to be masked by the input instruction (internal instruction code). The instruction mask IM_k defines "O(Zero)" to ~~the~~a bit to be masked, and also "1" to ~~the~~a bit not to be masked.

[00050] The instruction code IC_k defines the internal instruction code to be compared with the input instruction. The instruction code IC_k defines "O(Zero)" ~~to the~~as a bit to be masked, and also defines a certain value ~~to~~for the ~~bit~~bits not to be masked by the instruction mask IM_k. The group code GC_k defines the changed group code to be set newly, in the case that the group code agrees with a condition set by the instruction mask IM_k and the instruction code IC_k.

[00051] For example, as shown in Fig.4, when the ~~value~~values "11110000000", "10110000000", "101" are set ~~to~~for the instruction mask IM1, the instruction code IC1, and the group code GC1 respectively, each of the 4 high order ~~4-bit~~bits of the instruction mask IM1 is "1". When the value of the 4 high order ~~4-bit~~bits of the input instruction is compared with the value of the 4 high order ~~4-bit~~bits of the instruction code IC1, if ~~agreed~~in agreement, the group code will be changed into "101". In the case that the value of the 4 high order ~~4-bit~~bits of the input instruction is "1011", the group code will be changed into "101".

[00052] Similarly, when the ~~value~~values "11111000000", "11011000000", "110" are set to the instruction mask IM2, the instruction code IC2, and the group code GC2 respectively, the group code will be changed into "110" if the value of the 5 high order ~~5-bit~~bits of the input instruction is "11011".

[00053] The LUT44 shown in Fig.4 is just one example, and ~~does~~the present invention is not limited to it. ~~For~~In another example, ~~the~~an LUT, wherein the plural groups of the instruction mask and the instruction code and one changed group code are set as one group, and the group code will be changed according to plural input instructions including the previous input instruction, can be acceptable.

[00054] Fig.5 is a view showing one example of a group change determination circuit for determining whether the input instruction (internal instruction code) satisfies the rule for changing the group code shown in Fig.4, or not. This determination circuit is provided~~ to~~, for example, in the group change instruction decoder 37.

[00055] As shown in Fig.5, in the determination circuit, the logical multiplication operation between one bit IN_m ~~(~~, where "m" is a subscript, and an integer ~~number of~~between 0-10,~~ and the followings are the same condition)~~ in

the input instruction and one bit MB_m in the instruction mask MS_k is performed at a logical multiplication operation (AND) circuit 51 ~~in every~~for each corresponding bit. Further, ~~the~~an exclusive negative OR operation between the operation result of the AND circuit 51 and one bit IB_m in the instruction code CODE is performed at an exclusive negative OR operation (EX-NOR) circuit 52 ~~in every~~for each corresponding bit. By means of an AND circuit 53 connected ~~dependently~~in series, the logical multiplication operation of all operation results of ~~every~~each EX-NOR circuit 52 is carried out so as to output the operation result thereof as a selecting signal SEL.

[00056] ~~By~~With the determination circuit thus being configured, the selecting signal will be at a high-level only in the case ~~that~~where the input instruction agrees with the condition of the changing rule set by the instruction mask MS_k and the instruction code CODE~~, as~~. As a result, the changed group code set by the said changing rule will be selected.

[00057] According to the present embodiment as described above, in the parallel processor 10 having the plural processors 11, the instruction decoding portion 13 in each processor 11 adds the group code which is determined ~~by~~a the past instruction code to the instruction code which is read out of the instruction memory so as to generate the internal instruction code. Further, the instruction decoding portion 13 performs various kinds of controls, uniquely determining the instruction to be executed~~ uniquely~~, based on the internal instruction code, from among the plurality of instructions assigned to the instruction code which is read out.

[00058] Accordingly, the plural instructions correspond to one instruction code to express ~~every instruction by~~each of the instructions with a short instruction code length, so that the increase of the capacity required ~~to~~for the program memory holding the ~~said~~ instruction code can be lowered. Moreover, different kinds of processing can be performed with the same instruction code, according to the group code determined by ~~the~~a past instruction code~~, as~~. As a result, various and advanced processing can be executed ~~comparing the~~compared to hitherto ~~processor~~available processors. Besides, by setting the alias instruction to which an optional instruction is assigned, the instruction of other groups can be executed immediately by one instruction code~~ immediately~~.

[00059] In the above described embodiment, the instruction code is ~~set~~defined by an 8-bit code, and the group code is ~~set~~defined by a 3-bit code, however, ~~it~~this is ~~an preferable~~only a preferred example and the present invention ~~does~~is not limited ~~in~~

~~i t~~to this example.

[00060] The present embodiments are to be considered in all respects as illustrative and ~~no~~not restrictive, and all changes which come within the meaning and range of equivalency of the claims are therefore intended to be embraced therein. The invention may be embodied in other specific forms without departing from the spirit or essential characteristics thereof.

[00061] As described above, according to the present invention, without a need for different kinds of instruction codes corresponding to every instruction which can be executed in the processor, one instruction code corresponds to ~~the~~a plurality of instructions and each processor determines the instruction to be executed uniquely, based on the input history of instruction codes, from the plural instructions by the instruction code. Thus, every instruction can be expressed by short instruction code length so that the increase of memory capacity required to hold the instruction code at each processor is well lowered. Also, if the same instruction code is inputted, different kinds of instructions can be executed according to the input history of the instruction code, as a result, various and different kinds of processing can be executed by the plurality of processors in parallel.